NAG Library Routine Document

E02ALF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

E02ALF calculates a minimax polynomial fit to a set of data points.

2 Specification

```
SUBROUTINE E02ALF (N, X, Y, M, A, REF, IFAIL)
INTEGER            N, M, IFAIL
REAL (KIND=nag_wp) X(N), Y(N), A(M+1), REF
```

3 Description

Given a set of data points (x_i, y_i), for $i = 1, 2, \ldots, n$, E02ALF uses the exchange algorithm to compute an mth-degree polynomial

$$\mathrm{P}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m$$

such that $\max_i |\mathrm{P}(x_i) - y_i|$ is a minimum.

The routine also returns a number whose absolute value is the final reference deviation (see Section 5). The routine is an adaptation of Boothroyd (1967).

4 References

Boothroyd J B (1967) Algorithm 318 *Comm. ACM* **10** 801

Stieffel E (1959) Numerical methods of Tchebycheff approximation *On Numerical Approximation* (ed R E Langer) 217–232 University of Wisconsin Press

5 Arguments

1: N – INTEGER *Input*

On entry: n, the number of data points.

Constraint: $\mathrm{N} \geq 1$.

2: X(N) – REAL (KIND=nag_wp) array *Input*

On entry: the values of the x coordinates, x_i, for $i = 1, 2, \ldots, n$.

Constraint: $x_1 < x_2 < \ldots < x_n$.

3: Y(N) – REAL (KIND=nag_wp) array *Input*

On entry: the values of the y coordinates, y_i, for $i = 1, 2, \ldots, n$.

4: M – INTEGER *Input*

On entry: m, where m is the degree of the polynomial to be found.

Constraint: $0 \leq \mathrm{M} < \min(100, \mathrm{N} - 1)$.

5: A(M + 1) – REAL (KIND=nag_wp) array *Output*

On exit: the coefficients a_i of the minimax polynomial, for $i = 0, 1, \ldots, m$.

6: REF – REAL (KIND=nag_wp) *Output*

On exit: the final reference deviation, i.e., the maximum deviation of the computed polynomial evaluated at x_i from the reference values y_i, for $i = 1, 2, \ldots, n$. REF may return a negative value which indicates that the algorithm started to cycle due to round-off errors.

7: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value −1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value −1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, N = ⟨*value*⟩.
Constraint: N ≥ 1.

IFAIL = 2

On entry, M = ⟨*value*⟩.
Constraint: M < 100.

On entry, M = ⟨*value*⟩.
Constraint: M ≥ 0.

On entry, M = ⟨*value*⟩ and N = ⟨*value*⟩.
Constraint: M < N − 1.

IFAIL = 3

On entry, i = ⟨*value*⟩, X(i + 1) = ⟨*value*⟩ and X(i) = ⟨*value*⟩.
Constraint: X(i + 1) > X(i).

IFAIL = −99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = −399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

This is dependent on the given data points and on the degree of the polynomial. The data points should represent a fairly smooth function which does not contain regions with markedly different behaviours. For large numbers of data points (N $>$ 100, say), rounding error will affect the computation regardless of the quality of the data; in this case, relatively small degree polynomials (M $\ll \sqrt{\text{N}}$) may be used when this is consistent with the required approximation. A limit of 99 is placed on the degree of polynomial since it is known from experiment that a complete loss of accuracy often results from using such high degree polynomials in this form of the algorithm.

8 Parallelism and Performance

E02ALF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The time taken increases with m.

10 Example

This example calculates a minimax fit with a polynomial of degree 5 to the exponential function evaluated at 21 points over the interval $[0, 1]$. It then prints values of the function and the fitted polynomial.

10.1 Program Text

```
    Program e02alfe

!     E02ALF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: e02alf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: dxx, ref, s, t, xx
      Integer                          :: i, ifail, j, m, n, neval
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:), x(:), y(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: exp, real
!     .. Executable Statements ..
      Write (nout,*) 'E02ALF Example Program Results'

!     Skip heading in data file
      Read (nin,*)
```

```
      Read (nin,*) n, m, neval
      Allocate (a(m+1),x(n),y(n))

      Read (nin,*)(x(i),y(i),i=1,n)

      ifail = 0
      Call e02alf(n,x,y,m,a,ref,ifail)

      Write (nout,*)
      Write (nout,*) '   Polynomial coefficients'
      Write (nout,99998)(a(i),i=1,m+1)
      Write (nout,*)
      Write (nout,99997) '   Reference deviation = ', ref
      Write (nout,*)
      Write (nout,*) '  x     Fit      exp(x)   Residual'

!     The neval evaluation points are equispaced on [0,1].
      dxx = 1.0_nag_wp/real(neval-1,kind=nag_wp)

      Do j = 1, neval
        xx = real(j-1,kind=nag_wp)*dxx

        s = a(m+1)

        Do i = m, 1, -1
          s = s*xx + a(i)
        End Do

        t = exp(xx)
        Write (nout,99999) xx, s, t, s - t
      End Do

99999 Format (1X,F5.2,2F9.4,E11.2)
99998 Format (6X,E12.4)
99997 Format (1X,A,E10.2)
    End Program e02alfe
```

10.2 Program Data

```
E02ALF Example Program Data
 21     5   11             : N, M, NEVAL
  0.00      1.0000000000
  0.05      1.0512710964
  0.10      1.1051709181
  0.15      1.1618342427
  0.20      1.2214027582
  0.25      1.2840254167
  0.30      1.3498588076
  0.35      1.4190675486
  0.40      1.4918246976
  0.45      1.5683121855
  0.50      1.6487212707
  0.55      1.7332530179
  0.60      1.8221188004
  0.65      1.9155408290
  0.70      2.0137527075
  0.75      2.1170000166
  0.80      2.2255409285
  0.85      2.3396468519
  0.90      2.4596031112
  0.95      2.5857096593
  1.00      2.7182818285   : X, Y
```

10.3 Program Results

```
 E02ALF Example Program Results

    Polynomial coefficients
        0.1000E+01
        0.1000E+01
        0.4991E+00
        0.1704E+00
        0.3478E-01
        0.1391E-01

    Reference deviation =   0.11E-05

   x     Fit      exp(x)   Residual
 0.00   1.0000   1.0000  -0.11E-05
 0.10   1.1052   1.1052   0.97E-06
 0.20   1.2214   1.2214  -0.74E-06
 0.30   1.3499   1.3499  -0.92E-06
 0.40   1.4918   1.4918   0.30E-06
 0.50   1.6487   1.6487   0.11E-05
 0.60   1.8221   1.8221   0.46E-06
 0.70   2.0138   2.0138  -0.82E-06
 0.80   2.2255   2.2255  -0.84E-06
 0.90   2.4596   2.4596   0.88E-06
 1.00   2.7183   2.7183  -0.11E-05
```